ESC-TR-2003-070

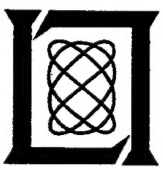
Project Report
PCA-IRT-2

Preliminary Design Review: GMTI Processing for the PCA Integrated Radar-Tracker Application

A.I. Reuther

8 April 2002
Issued 6 February 2004

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS



Prepared for the Defense Advanced Research Projects Agency
under Air Force Contract F19628-00-C-0002.

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by DARPA/ITO under Air Force Contract F19628-00-C-0002.

The ESC Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Gary Tutungian
Administrative Contracting Officer
Plans and Programs Directorate
Contracted Support Management

Massachusetts Institue of Technology
Lincoln Laboratory

Preliminary Design Review: GMTI Processing for the PCA Integrated Radar-Tracker Application

A.I. Reuther
Group 102

Project Report PCA-IRT-2

8 April 2002
Issued 6 February 2004

Lexington Massachusetts

ABSTRACT

This report describes ground moving target indicator (GMTI) processing to be done for the PCA (polymorphous computing architecture) integrated radar-tracker application. GMTI processing of the raw radar data is done to extract targets; target reports are passed on to the tracker part of the integrated radar-tracker application.

The GMTI processing described in this report is used to minimize the effects of interference patterns and ground clutter and to recognize and extract targets on the ground. The radar is presumed to be using a wideband signal for improved resolution. This signal is decomposed via subband filtering into many narrowband signals for easier processing. Each of these subbanded data sets must then be processed as a distinct radar data cube. For each data set, adaptive processing is used to null interference patterns and ground clutter noise. These operations incur a high computational cost. Subband filtering requires many parallel modulators and filters, while adaptive processing requires the calculation of an adaptive filter (involving a QR decomposition and forward/backsolving) at run time for each subband data cube.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

This report describes ground moving target indicator (GMTI) processing to be done for the PCA (polymorphous computing architecture) integrated radar-tracker application. GMTI processing of the raw radar data is done to extract targets; target reports are passed on to the tracker part of the integrated radar-tracker application.

The GMTI processing described in this report is used to minimize the effects of interference patterns and ground clutter and to recognize and extract targets on the ground. The radar is presumed to be using a wideband signal for improved resolution. This signal is decomposed via subband filtering into many narrowband signals for easier processing.[1] Each of these subbanded data sets must then be processed as a distinct radar data cube. For each data set, adaptive processing is used to null interference patterns and ground clutter noise. These operations incur a high computational cost. Subband filtering requires many parallel modulators and filters, while adaptive processing requires the calculation of an adaptive filter (involving a QR decomposition and forward/backsolving) at run time for each subband data cube.

This report describes the key features and elements for a preliminary design review of the GMTI radar processing component of the integrated moving target indicator/tracker application. This report includes:

- a high-level description of the GMTI radar processing chain
- specifications of the input and output for each of the GMTI radar processing chain elements
- detailed descriptions of each of the GMTI radar processing chain elements

1.1 THE GMTI RADAR PROCESSING CHAIN

Before delving into the details of each of the elements of the processing chain, a description of the entire GMTI processing chain is in order. The processing chain is illustrated in Figure 1. In this figure, the small numbers in the upper right corner of the boxes refer to the section in which that stage is discussed in greater detail. The shaded box with bold labels is described in [1].

---

[1] A wideband signal is a signal in which the bandwidth is a significant fraction of the center frequency, while a narrowband signal is one in which the bandwidth is an insignificant fraction of the center frequency. The determination of what constitutes a significant fraction of the center frequency is application specific. As this is a wideband radar, the assumptions made for narrowband processing break down, leading to dispersion in the beams that are formed.

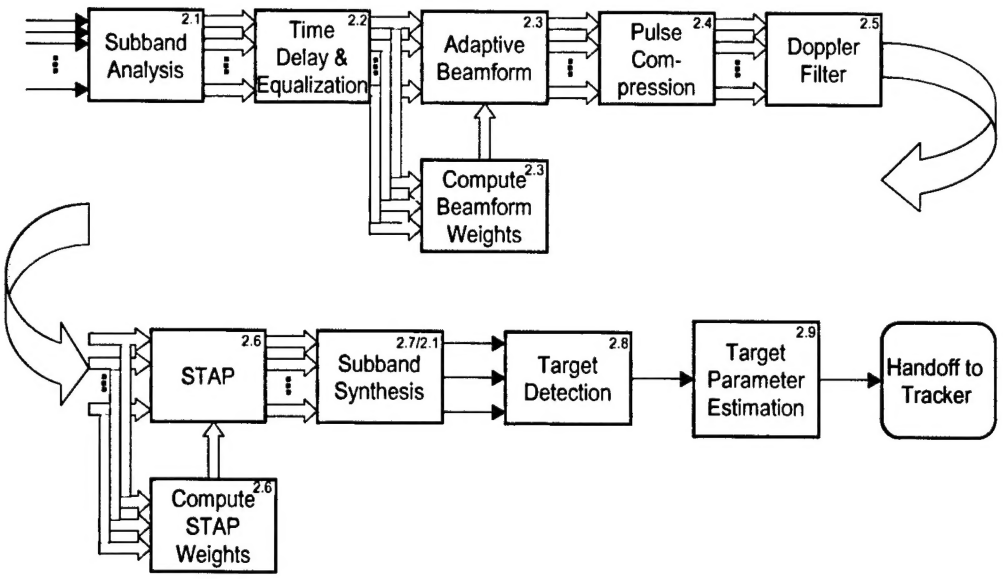


Figure 1. GMTI processing chain.

Before the radar data is processed by the GMTI stages, a radar signal consisting of a series of pulses comprising a coherent processing interval (CPI) has been transmitted. The pulse repetition interval (PRI) determines the time interval between transmitted pulses, and the number of pulses per CPI is given as N_{pri}. The signal reflects off of targets, the ground, and other entities and is received by the radar antenna. The antenna is an array of N_{ch} sensors, which are also called channels. These channel sensor inputs are digitized by A/D converters at a rate of N_{rg} samples per PRI. The digitization rate alludes to the relationships between the range dimension and the PRI dimension: the range dimension is referred to as the fast time dimension, while the PRI dimension is the slow time dimension. This $N_{ch} \times N_{rg} \times N_{pri}$ digitized data cube of samples is the input to the GMTI processing chain.

The GMTI processing chain consists of nine stages: subband analysis (decomposing); time delay and equalization; adaptive beamforming; pulse compression; Doppler filtering; space-time adaptive processing (STAP); subband synthesis (recombining); detection; and estimation. This processing chain then reports the resulting targets to the tracker.

- The *subband analysis stage* breaks the wideband radar signal into a series of narrowband signals to prevent beam dispersion which facilitates more efficient processing.
- The *time delay and equalization stage* compensates for differences in the transfer function between channel sensors.

- The *adaptive beamforming stage* transforms the filtered data into the beam-space domain to allow detection of target signals coming from a particular set of directions of interest while filtering out spatially-localized interference.
- The *pulse compression stage* filters the data to concentrate the signal energy of a relatively long transmitted radar pulse into a relatively short pulse response.
- The *Doppler filter stage* processes the data so that the radial velocity of targets relative to the platform can be determined.
- The *STAP stage* is a second beamforming stage which removes further interference and ground clutter interference.
- The *subband synthesis stage* stitches the processed subband signals back together, reversing the work that the subband analysis stage had conducted on the unprocessed signal.
- The *detection stage* uses constant false-alarm rate (CFAR) detection to compare a radar signal response to its surrounding signal responses to determine whether a target is present and uses target grouping to eliminate multiple target reports that are actually just one target.
- The *estimation stage* estimates target positions in order to pass them to the tracking algorithms.
- Finally, the target report is passed to the tracking algorithms which are described in [1].

1.2 GMTI CONTROL INPUT DATA

The GMTI processing function requires a control input which is a Matlab structure. The names of the elements, the variables they correspond to in this report, and the descriptions of each element are given in Table 1. This structure must be passed to the GMTI function when it is called.

TABLE 1

Control Input Parameters for GMTI Processing

Parameter	Element Name	Description
N_{ch}	Nch	Number of radar sensor channels
N_{rg}	Nrg	Number of range gates per radar data cube
N_{pri}	Npri	Number of pulse repetition intervals per radar data cube
N_{sub}	Nsub	Number of subbands for processing
N_{srg}	Nsrg	Number of subband range gates in radar data cube
N_{dwn}	Ndwn	Down-sample rate for the subband analysis stage, and conversely, the up-sample rate for the subband synthesis stage
N_{sfd}	Nsfd	Number of finite impulse response (FIR) filter taps in the subband analysis down-sampling low-pass filter
N_{sfu}	Nsfu	Number of FIR filter taps in the subband synthesis up-sampling low-pass filter
N_{tde}	Ntde	Number of FIR filter taps in the time delay and equalization filter
α	alpha	Diagonal loading factor in the adaptive beamforming stage
N_{pc}	Npc	Number of filter taps in the pulse compression FIR filter
N_{bm}	Nbm	Number of beams in the radar data cube after the adaptive beamforming stage
N_{dop}	Ndop	Number of Doppler bins for Doppler filtering
N_{stag}	Nstag	Number of PRI staggers produced by Doppler filtering for use in the STAP stage
N_{ts}	Nts	Number of training samples for the STAP stage
N_{cnb}	Ncnb	Number of clutter nulled beams in the radar data cube after the STAP stage
N_{cfar}	Ncfar	Number of range gates used in forming the noise estimate from each side of the cell under test
G	G	Number of guard range cells on both sides of the cell under test
μ	mu	Normalized target power threshold

Most of the control input parameters presented in the table above are also shown in tables at the end of the stage section in which they are used.

Beyond the control input parameters, there are a number of data product inputs, as shown in Table 2.

TABLE 2

Data Product Inputs for GMTI Processing

Parameter	Element Name	Description
$h_s(n)$	`hs`	Coefficient vector of subband analysis low-pass filter
$f_s(n)$	`fs`	Coefficient vector of subband synthesis low-pass filter
τ_{bm}	`Tbm`	Vector of range gate indices for adaptive beamforming stage training samples
h_{pc}	`hpc`	Coefficient vector of the pulse compression filter
τ_{stag}	`Tstag`	Vector of space-time snapshot indices for the training data for space-time adaptive processing

1.3 GMTI INPUT DATA

The input data to the GMTI radar processing chain is a series of three-dimensional radar data cubes whose dimensions are channel, range, and pulses (see Figure 2). The collection of the samples of the radar data cubes is explained in Section 1.1. Individual elements of the radar data cube C are referred to as $C(i, j, k)$, where i is the channel index, j is the range index, and k is the pulse index.

The series of radar data cubes will be indexed using a cell array in Matlab:[2]

```
input_data{1} = cube1;
input_data{2} = cube2;
input_data{3} = cube3;
etc.
```

[2] Items in `Courier font` correspond to functions or variables in the Matlab code.

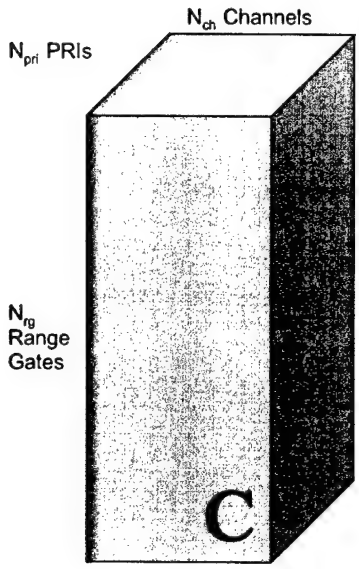


Figure 2. Radar data cube, the input data to the GMTI processing.

1.4 GMTI OUTPUT DATA

The tracker receives the following data on each target:

- target power or signal-to-noise ratio (SNR)
- target azimuth (in degrees or radians)
- target range (in meters or kilometers)
- target radial velocity (in m/s or km/h)
- target time stamp

The tracker is described in [1].

2. FUNCTIONAL DESCRIPTION

In this section, each of the processing stages are described in greater detail. For convenience, Figure 3 shows the GMTI processing chain with the sizes of the passed parameters.

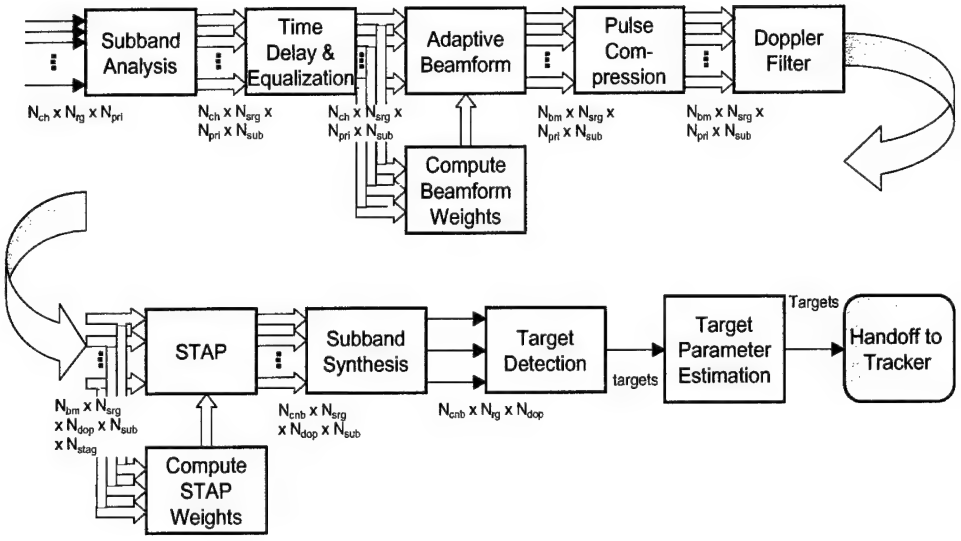


Figure 3. GMTI processing chain with passed parameter sizes.

2.1 SUBBAND ANALYSIS (AND SYNTHESIS)

Input: Wideband radar data cube of size $N_{ch} \times N_{rg} \times N_{pri}$.

Output: Narrowband N_{sub} subband data cubes of size $N_{ch} \times N_{srg} \times N_{pri}$.

In order to get greater resolution in recognizing targets, it is beneficial to use wideband radar pulses. However, channel equalization, pulse compression, Doppler filtering, and adaptive nulling/beamforming must be performed on a narrowband signal because the assumptions made for narrowband processing break down in a wideband scenario, leading to dispersion in the beams that are formed. (Compensation for beam dispersion makes a system far more complex.) Therefore, the processing chain must involve decomposing the wideband signal into many downsampled narrowband signals, processing each subband individually, and then recombining the subband signals prior to performing target detection [6]. The subband filter is implemented with a classic polyphase filter [5]. Since subband analysis and synthesis are duals of each other, both stages will be discussed within this section.

Subband analysis consists of modulating to baseband the incoming signal into a series of subband signals, low-pass filtering each of these subband signals with $h_s(n)$—an N_{sfd}-tap filter—and downsampling the subsequent filtered signals. The inputs to this function are the range vectors of the input radar data cube: $x = C(i, :, k)$, where $x(n) = C(i, n, k)$. The modulation is performed by multiplying each sample by a frequency down-shifting value $W_{N_{sub}}^{-kn}$, where $W_{N_{sub}} = e^{(j2\pi)/N_{sub}}$ and $0 \le k < N_{sub}$. The low-pass filtering is conducted so that signal aliasing is minimized in the down-sampling step. Down-sampling is conducted by simply extracting one out of every N_{dwn} samples. When these operations are completed, the output is a set of N_{sub} subband data cube of size $N_{ch} \times N_{srg} \times N_{pri}$, where $N_{srg} = N_{rg}/N_{dwn}$. This transformation is depicted in Figure 4. In Matlab notation, these data cubes will be referred to as $C(i, j, k, s)$, where i is the channel index, j is the subband range index, k is the PRI index, and s is the subband index. Often N_{sub} and N_{dwn} are chosen such that $N_{sub} = N_{dwn}$; this means that the total number of samples is preserved between the input data cube and the N_{sub} subband data cubes.

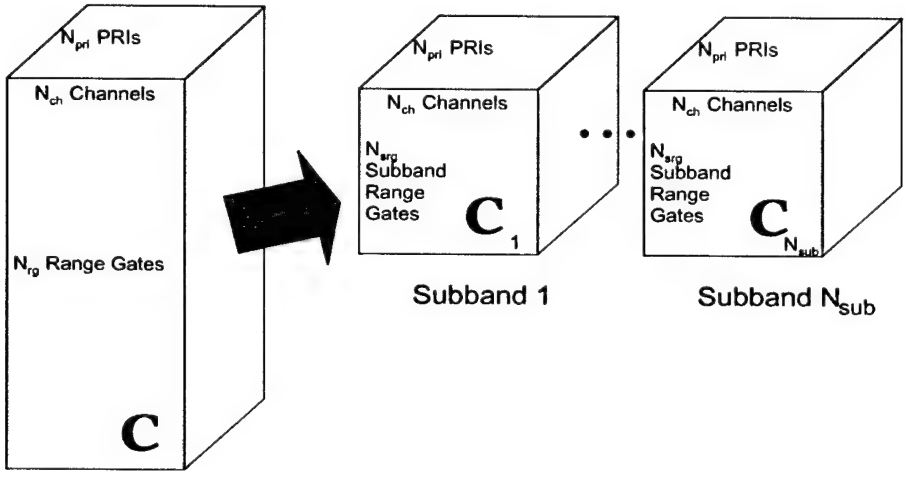


Figure 4. Data size transformation in subband analysis stage.

Conversely, subband synthesis involves the dual: up-sampling the incoming subband signals, low-pass filtering the up-sampled subband signals with $f_s(n)$—an N_{sfu} N-tap filter—modulating the subband signals back up to their original frequencies, and recombining the subbands by superposition addition. Upsampling is performed by inserting $N_{dwn} - 1$ zero-valued samples between each sample. Again, low-pass filtering is conducted to minimize signal aliasing in the up-sampling step. The modulation is performed by multiplying each sample by a frequency up-shifting value $W_{N_{sub}}^{kn}$, where $W_{N_{sub}} = e^{(j2\pi)/N_{sub}}$ and $0 \le k < N_{sub}$. The subband analysis and synthesis processes are depicted in Figure 5.

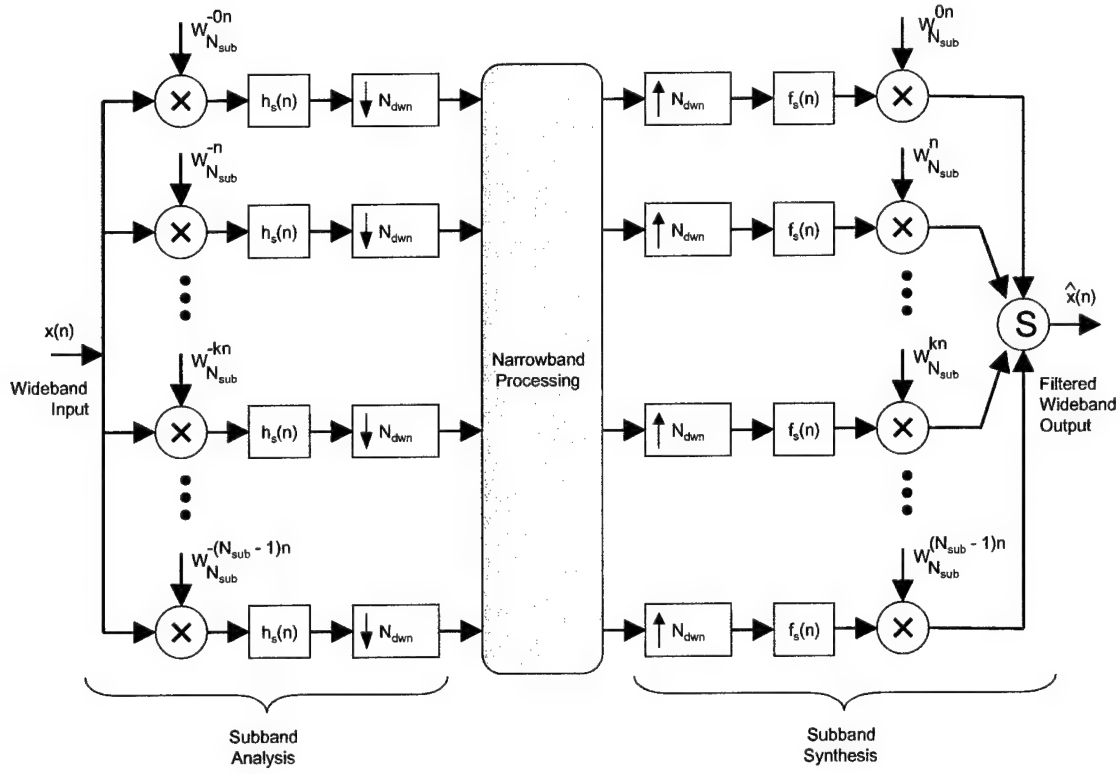


Figure 5. GMTI subband filtering stage process.

This structure is required for each channel of the system. There is a great deal of redundancy within each channel's subband processing, especially in the low-pass filters. Each of the $h_s(n)$ downsampling and $f_s(n)$ upsampling low-pass filters have the same coefficients. Therefore, some novel signal processing "tricks" can be implemented to reduce the operations count [7],[8]. However, these methods are beyond the scope of this report.

The data parameters of the subband filtering stages are listed in Table 3.

TABLE 3

Data Parameters of the Subband Filtering Stages

Name	Description
N_{sub}	Number of subbands for processing
N_{sfd}	Number of taps for subband down-sampling low-pass filter
N_{sfu}	Number of taps for subband up-sampling low-pass filter

2.2 TIME DELAY AND EQUALIZATION

Input: N_{sub} subband data cubes of size $N_{ch} \times N_{srg} \times N_{pri}$.

Output: N_{sub} subband data cubes of size $N_{ch} \times N_{srg} \times N_{pri}$.

The time delay and equalization stage is applied to each subband, and it uses a finite impulse response (FIR) filter on each range vector for each channel, each pulse, and each subband to compensate for signal differences between channel sensors. That is, in Matlab notation, it operates on each $x = C(i, :, k, s)$, where i is the channel index, k is the PRI index, and s is the subband index. In its simplest form, a single tap adjustment can be folded into the digital beamforming weights; however, more complex equalization requires using programmable FIR filters. The FIR filter has filter coefficients, $h_{tde}[l]$, $0 \le l \le N_{tde} - 1$. The output of the filter, the vector y, is the convolution of the filter h_{tde} with the input x:

$$y[j] = \sum_{l=0}^{N_{tde}-1} x[j-l]h_{tde}[l], \text{ for } j = 0, 1, \ldots N_{rg} \quad . \tag{1}$$

Depending on the number of filter coefficients, N_{tde}, this filter can be implemented using fast convolution with FFTs [4] or by explicitly convolving the input signal and the filter coefficients as depicted in the equation above.

The time delay and equalization stage parameter is listed in Table 4.

TABLE 4

Time Delay and Equalization Stage Parameter

Name	Description
N_{tde}	Number of taps in the time delay and equalization filters

2.3 ADAPTIVE BEAMFORMING

Input: N_{sub} subband data cubes of size $N_{ch} \times N_{srg} \times N_{pri}$.

Output: N_{sub} subband data cubes of size $N_{bm} \times N_{srg} \times N_{pri}$.

The adaptive beamforming stage transforms the filtered data into the beam-space domain to allow detection of target signals coming from a particular set of directions of interest while filtering out jamming (spatially-localized) interference. The filtering is executed for each subband radar data cube within each CPI.

Again, we follow through the execution of this stage for just one subband radar data cube, namely the sth subband. To perform adaptive beamforming, a clutter-free data matrix $A^{(s)}$ is extracted from $C^{(s)}$.

Typically, the first PRI signal of a CPI is a receive-only PRI; that is, no signal is transmitted for the first PRI, and only the jamming signals are sensed by the radar receiver. This means using the channel columns samples from the range and pulse set; in Matlab terms, it is *C(:, j, k, s)* for a given set of *j's* and *k=1*. A total of $5N_{ch}$ channel column samples are taken, and the choice of which range samples to use is given by the $5N_{ch}$-length index vector τ_{bm}. Thus, $A^{(s)}$ is an $N_{ch} \times 5N_{ch}$ matrix.

In adaptive processing, the goal is to compute columns of the beamforming matrix, $W_{bm}^{(s)}$ (of size $N_{ch} \times N_{bm}$), by solving $w_m = (\hat{R}^{(s)})^{-1} v_m$, where w_m is the m-th column of $W_{bm}^{(s)}$, $\hat{R}^{(s)}$ is the estimated covariance matrix, and v_m is the m-th column of the steering matrix, $V^{(s)}$. To this end, the estimated covariance matrix, $\hat{R}^{(s)}$, is computed from the clutter-free data matrix added to the diagonal loading factor, α, multiplied by an $N_{ch} \times N_{ch}$ identity matrix. $\hat{R}^{(s)}$ is then a $N_{ch} \times N_{ch}$ matrix as shown in (2).

$$\hat{R}^{(s)} = \frac{1}{5N_{ch}} A^{(s)} (A^{(s)})^H + \alpha I \quad . \tag{2}$$

The m-th column of the steering matrix, $V^{(s)}$, is computed as

$$v_{m,s} = \left[b_0 e^{-jd_{s,m}}, b_1 e^{-j(a_{s,m} - d_{s,m})}, \ldots, b_{(N_{ch}-1)} e^{-j((N_{ch}-1)a_{s,m} - d_{s,m})} \right]^T \quad , \tag{3}$$

where $a_{s,m}$ and $d_{s,m}$ are provided for each subband and each $A^{(s)}$, and the b_i weights are static and determined as part of the radar design. The range for m is $0 \le m < N_{bm}$, so $V^{(s)}$ is an $N_{bm} \times N_{ch}$ matrix.

Now the m-th column of the beamforming, interference-nulling matrix $W_{bm}^{(s)}$ is computed as:

$$w_m = \frac{(\hat{R}^{(s)})^{-1} v_m}{v_m^H (R^{(s)})^{-1} v_m} \quad . \tag{4}$$

The denominator of Equation (2) is a scalar normalization factor. The above equation is referred to as a Wiener-Hopf equation for calculating adaptive filter weights. While this explains the mathematics behind calculating the interference-nulling matrix, calculating the inverse of $\hat{R}^{(s)}$ is not used due to numerical stability issues. Instead LQ-decomposition and forward- and back-substitution is used to calculate each of the w_m vectors.

As explained in Appendix A, define $x' = \left[x_T \; I\sqrt{\alpha} \right]$, a 1 by $1 \times 6N_{ch}$ vector. With $(x')(x')^H = \tilde{R}$, $L' = lqhouse((x')(x')^H)$; L' is a lower triangular $N_{ch} \times N_{ch}$ matrix. Now using t_m as a temporary vector of size N_{ch} by 1, we can solve $L' t_m = v_m$ for t_m using forward substitution and $(L')^H w'_m = t_m$ for w'_m using back-substitution. Finally, compute $w_m = w'_m / ((t_m)^H t_m)$ to take into account the denominator scalar normalization factor.

Refer to Appendix A for a deeper explanation of this method of solving Wiener-Hopf equation.

Finally, $W_{bm}^{(s)}$ is used to beamform the radar data cube. Using Matlab notation in defining the matrix $X_k^{(s)} = C(:, :, k, s)$ with size $N_{ch} \times N_{srg}$, the matrix $Y_k^{(s)}$ as the output of the beamforming,

$$Y_k^{(s)} = (W_{bm}^{(s)})^H X_k^{(s)} \quad . \tag{5}$$

Thus, the matrix $Y_k^{(s)}$ is of size $N_{bm} \times N_{srg}$. Figure 6 shows a conceptual data cube and its partitioning for this operation.

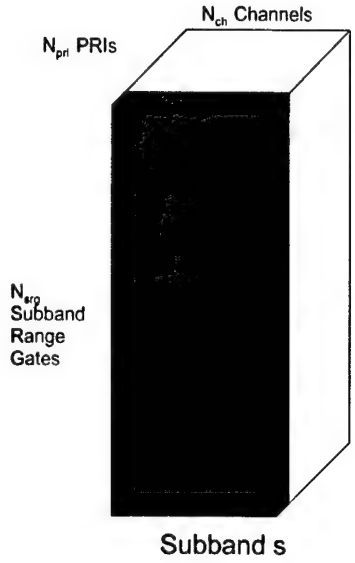


Figure 6. Data partitioning of input data cube for adaptive beamforming stage.

The data parameters of the adaptive beamforming stage are listed in Table 5.

TABLE 5
Adaptive Beamforming Stage Parameters

Name	Description
α	Diagonal loading factor
τ_{bm}	Vector of range gate indices for adaptive beamforming stage training samples

2.4 SUBBAND PULSE COMPRESSION

Input: N_{sub} subband data cubes of size $N_{bm} \times N_{srg} \times N_{pri}$.

Output: N_{sub} subband data cubes of size $N_{bm} \times N_{srg} \times N_{pri}$.

The pulse compression stage filters the data to concentrate the signal energy of a relatively long transmitted radar pulse into a relatively short pulse response. This compression is accomplished by applying a FIR filter operation to the range data for each pulse and channel within each subband radar data cube. That is, in Matlab notation, it operates on each $x = C(i, :, k, s)$, each of which are of length N_{srg}; this partitioning is illustrated in Figure 7. The FIR filter has coefficients $h_{pc}[l]$, $l \in \{0 \ldots N_{pc}-1\}$. The output of the filter, the vector y of length N_{srg}, is the convolution of the filter h_{pc} with the input x:

$$y[j] = \sum_{l=0}^{N_{pc}-1} x[j-l]h_{pc}[l], \text{ for } j = 0, 1, \ldots N_{rg} \quad . \tag{6}$$

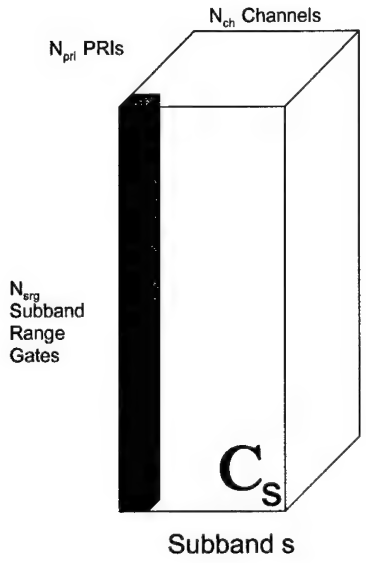


Figure 7. Data partitioning of input data cube for pulse compression stage.

Unlike the time delay and equalization stage, the number of taps in the pulse compression FIR filter is usually much larger than four or so taps. Therefore, it is usually more efficient to implement this convolution with the fast convolution technique using FFTs, possibly even considering an overlap-and-save method [4].

The subband pulse compression stage data parameter is listed in Table 6.

TABLE 6

Subband Pulse Compression Stage Parameters

Name	Description
N_{pc}	Number of taps in the subband pulse compression filter

2.5 DOPPLER FILTERING

Input: N_{sub} subband data cubes of size $N_{bm} \times N_{srg} \times N_{pri}$.

Output: N_{sub} subband data cubes of size $N_{bm} \times N_{srg} \times N_{dop} \times N_{stag}$.

The Doppler filter stage processes the data so that the radial velocity of targets relative to the platform can be determined. Doppler filtering accomplishes this using a straightforward FFT applied to the pulse data for each subband range gate and beam. A further step made in the Doppler filtering which aids the STAP stage in filtering out ground clutter involves taking staggered pulse sample sets; N_{stag} is the number of pulse staggers. Using g ($1 \le g \le N_{stag}$) as the index of the stagger number, the staggers generated in Matlab notation are $y_{i,j,g} = C(i, j, g:(N_{pri}-N_{stag}+g), s)$; $y_{i,j,g}$ is a vector of length N_{pri}. For example, with $N_{stag} = 3$ the first input stagger vector would include the first pulse sample up to the $N_{pri}-2$nd pulse sample, the second input stagger vector would include the second pulse sample up to the $N_{pri}-1$st pulse sample, and the third input stagger vector would include the third pulse sample up to the N_{pri} th (last) pulse sample. This is illustrated in Figure 8. With $N_{stag} = 1$, there is only one stagger which is the entire set of pulse samples for a given beam and subband range gate.

The output of the Doppler filter is

$$z_{i,j,g,s} = FFT(y_{i,j,g,s}) \quad , \tag{7}$$

for each subband, beam, subband range, and stagger. Each $z_{i,j,g,s}$ is a vector of size N_{dop}.

The parameters of the Doppler processing stage are listed in Table 7.

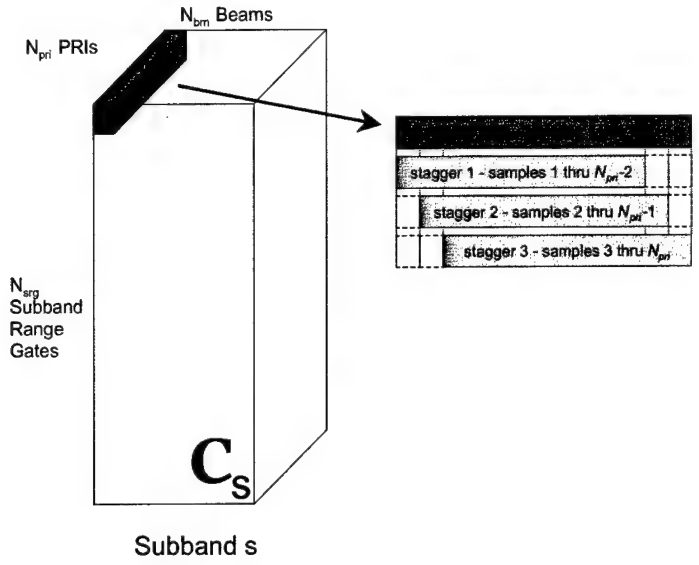


Figure 8. Data partitioning showing three staggers of input data cube for Doppler filtering stage.

TABLE 7

Doppler Processing Stage Parameters

Name	Description
N_{stag}	Number of pulse staggers

2.6 SPACE-TIME ADAPTIVE PROCESSING (STAP)

Input: N_{sub} subband data cubes of size $N_{bm} \times N_{srg} \times N_{dop} \times N_{stag}$.

Output: N_{sub} subband data cubes of size $N_{cnb} \times N_{srg} \times N_{dop}$.

The STAP stage is a second beamforming stage which removes further radar spatially-localized interference and ground clutter (space- and time-distributed) interference. Using the $z_{i,j,g}$ outputs from the Doppler filtering stage,

$$x_{j,k,g} = \begin{bmatrix} z_{0,j,g}(k) \\ \cdots \\ z_{N_{bm}-1,j,g}(k) \end{bmatrix} . \tag{8}$$

The size of $x_{j,k,g}$ is $N_{bm} \times 1$, and it is one stagger of samples. Next, the staggers are stacked upon each other to get a space-time snapshot:

$$x_{j,k} = \begin{bmatrix} x_{j,k,1} \\ \cdots \\ x_{j,k,N_{stag}+1} \end{bmatrix} \tag{9}$$

for the test cells, which makes $x_{j,k}$ is $N_{bm}N_{stag} \times 1$. We will use these vectors of size $N_{bm}N_{stag} \times 1$ as our input data to the STAP filter, but we shall first build the adaptive filter matrix. Defining

$$v_o(r) = e^{\frac{-j2\pi r}{N_{ch}}}, r = 0, \ldots, N_{ch} - 1 \quad , \tag{10}$$

and using the adaptive beamforming matrix of the subband data cube currently being processed, $W_{bm}^{(s)}$ of size $N_{ch} \times N_{bm}$, define the steering vector v, with length N_{bm}, as:

$$v = (W_{bm}^{(s)})^H v_o \quad . \tag{11}$$

The steering vector v must be expanded to become size $N_{bm}N_{stag} \times N_{cnb}$. Refer to the Matlab code for the actual operations; for the sake of discussion, we can define the $N_{bm}N_{stag} \times N_{cnt}$ dimensioned steering matrix $V = repmat(v, N_{stag}, N_{cnb})$. To simplify the discussion, let us define $v_b = V(:, b)$, where $1 \le b \le N_{cnb}$.

Now we will choose the space-time snapshots for the training data from this set:

$$\tau_{stap} = \left\{ x_{\tau_1,k}, \ldots, x_{\tau_{N_{ts}},k} \right\} \quad , \tag{12}$$

and with it build the interference estimate covariance matrix:

$$R_k = \frac{1}{N_{ts}} \sum_{u=1}^{N_{ts}} x_{\tau_u,k} x_{\tau_u,k}^H \quad . \tag{13}$$

Now the corresponding adaptive weights are calculated as:

$$w_{k,b} = R_k^{-1} v_b \quad . \tag{14}$$

As in adaptive beamforming, the above equation is referred to as the Wiener-Hopf equation for calculating adaptive filter weights. While this explains the mathematics behind calculating the

interference-nulling matrix, calculating the inverse of R_k is not used. Instead LQ-decomposition is used to calculate each of the $w_{k,b}$ vectors.

As explained in Appendix A, calculate $L_k = lqhouse(R_k)$; L_k is a lower triangular $N_{bm}N_{stag} \times N_{bm}N_{stag}$ matrix. Now using $t = L_k^H w_{k,b}$ as a temporary vector of size $N_{bm}N_{stag} \times 1$, we can solve $L_k t = v_b$ for t using forward-substitution and $L_k^H w_{k,b} = t$ for $w_{k,b}$ using back-substitution. Solving this for every $b = 1 \ldots N_{cnb}$ provides $W_k = \left[w_{k,1} \ldots w_{k,N_{cnb}} \right]$ for Doppler bank k; W_k is a matrix of size $N_{bm}N_{stag} \times N_{cnb}$. Of course all of these calculations must be done for each Doppler bank, $k = 1 \ldots N_{dop}$.

Refer to Appendix A for an explanation of this method of solving the Wiener-Hopf equation.

Finally, the adaptive weights vectors are applied to the snapshots $x_{j,k}$ of size $N_{bm}N_{stag} \times 1$:

$$y_{j,k} = W_k^H x_{j,k} \quad . \tag{15}$$

The resulting vector $y_{j,k}$ is of size $N_{cnb} \times 1$ and must be calculated for each $j = 1 \ldots N_{srg}$ and $k = 1 \ldots N_{dop}$.

The STAP stage data parameters are listed in Table 8.

TABLE 8

Space-Time Adaptive Processing Stage Parameters

Name	Description
N_{stag}	Number of pulse staggers
N_{ts}	Number of training data set vectors
τ_{stag}	Vector of space-time snapshot indices for the training data set for space-time adaptive processing

2.7 SUBBAND SYNTHESIS (COMBINING)

Input: Narrowband N_{sub} subband data cubes of size $N_{cnb} \times N_{srg} \times N_{dop}$.

Output: Processed wideband radar data cube of size $N_{cnb} \times N_{rg} \times N_{dop}$.

Since the subband analysis and synthesis are signal processing duals of one another, the subband synthesis stage was covered in Section 2.1.

2.8 TARGET DETECTION

Input: Processed radar data cube of size $N_{cnb} \times N_{rg} \times N_{dop}$.

Output: Coordinates of detected targets in radar data cube.

Though there are many techniques for detecting targets in the processed radar data cube including various integrators and hard limiters, this application shall use a version of constant false alarm rate (CFAR) detection with three-dimensional grouping. The data parameters of the target detection stage are listed in Table 9.

2.8.1 CFAR Detection

During CFAR detection, a local noise estimate is computed from the $2N_{cfar}$ range gates near the cell *C(i, j, k)* under test. A number of guard gates *G* immediately next to the cell under test will not be included in the local noise estimate (this number does not affect the throughput). The range cells involved in calculating the noise estimate for a particular Doppler bin and beam are shown in Figure 9. For each cell *C(i, j, k)*, the value of the noise estimate *T(i, j, k)* is calculated as:

$$T(i,j,k) = \frac{1}{2N_{cfar}} \sum_{l=G+1}^{G+N_{cfar}} |C(i,j+l,k)|^2 + |C(i,j-l,k)|^2 \quad . \tag{16}$$

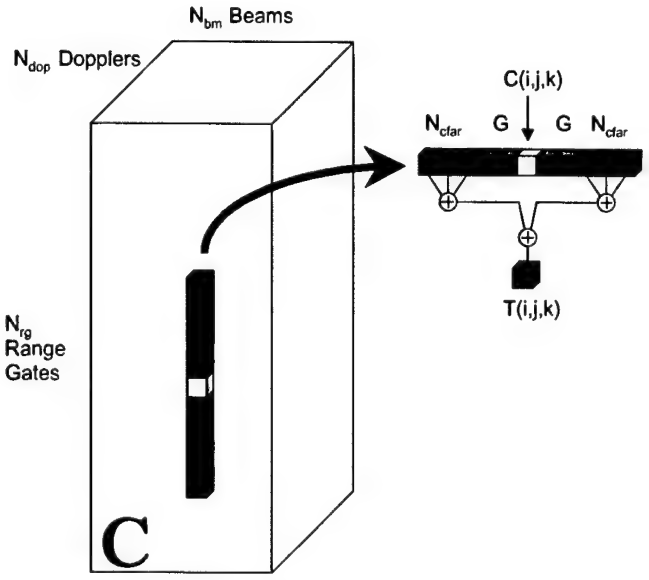


Figure 9. Data partitioning of input for CFAR detection stage.

At the boundaries when $j-(G+N_{cfar}-1)<1$ or $j+(G+N_{cfar})>N_{rg}$, only the side range gate cells that are within the data cube are included, and the denominator of the leading scaling factor is decreased to reflect the number of side range gate cells that were actually used for the calculation. For each cell *C(i, j, k)*, the quantity $|C(i,j,k)|^2/T(i,j,k)$ is calculated; this represents the normalized power in the cell under test. If this normalized power exceeds a threshold μ, the cell is considered to contain a target. This threshold can be adaptively determined, but for this application, we shall set it to a constant. The only cells in the data cube that are processed by the following stages are those where a target has been detected.

2.8.2 Three-dimensional Grouping

During this step, detection in adjacent cells are grouped to avoid having multiple detections associated with the same target. The power of each target detected by the CFAR algorithm will be compared to the power of each cell in the $3\times3\times3$ cube centered on that cell. Therefore, each cell under test will be compared to 26 other cells ($27-1$, where $27 = 3^3$ and 1 cell is the cell under test, the cell in the middle of the cube). If the detection's raw power is the local maximum, it is retained. Otherwise, the detection is discarded; it is considered grouped with the detection in the cell with the higher power.

TABLE 9

Target Detection Stage Parameters

Name	Description
N_{cfar}	Number of range gates used in forming the noise estimate from each side of the cell under test
G	Number of guard cells on both sides of the cell under test
μ	Normalized target power threshold

2.9 TARGET PARAMETER ESTIMATION

Input: Coordinates of detected targets in radar data cube.

Output: A target report.

This stage of the GMTI processing packages the target parameters for the target report. As presented in Section 1.4, a target report consists of the following data for each target:

- target power or signal-to-noise ratio (SNR)
- target azimuth (in degrees or radians)
- target range (in meters or kilometers)
- target radial velocity (in m/s or km/h)

- target time stamp

The target power is the quantity $|C(i,j,k)|^2/T(i,j,k)$ for each target cell *C(i, j, k)* which was calculated by the CFAR detection. The target azimuth is determined by the beam index number and the platform position and heading. We shall assume a temporarily stationary platform, so it will suffice to just use the beam index number. The target range is determined by the range gate index number and the platform altitude, grazing angle, and azimuth angle. Again, we shall assume a temporarily stationary platform, so it will suffice to just use the range gate index number. The target radial velocity is determined by the Doppler index number and the speed and heading of the platform, but with the assumed temporarily stationary platform, passing the Doppler index number will suffice. The target time stamp will be reported as the index to the data cube from which this target was determined.

3. MATLAB IMPLEMENTATION

The GMTI processing chain will be initialized with a function with the following signature:

```
GMTI_params = GMTI_initialize();
```

This function will load control parameters outlined in Section 1.2 into the structure `GMTI_params`.

The GMTI processing chain will be invoked with a function with the following signature:

```
targets = GMTI(input_data, GMTI_params);
```

This function call passes the aforementioned `GMTI_params` and the `input_data` outlined in Section 1.3, and it returns the target structure outlined in Section 1.4.

In order to keep memory usage to a reasonable level, the `GMTI` function will process only one radar data cube at a time, and within that radar data cube, it will process only one subband radar data cube at a time. Each of the stages depicted in Figure 1 will have its own function call within the `GMTI` function.

APPENDEX A
SOLVING THE WIENER-HOPF EQUATION

The Wiener-Hopf equation is used in both the adaptive beamforming and STAP stages. The equation $w = \tilde{R}^{-1} v$ defines an adaptive filter vector, w, as the product of the inverted estimated noise covariance matrix, $\tilde{R}$, and the nonadaptive steering vector, v. $\tilde{R}$ is composed using a set of training vectors, x_T, such that $\tilde{R} = (x_T)(x_T)^H$.

However, $\tilde{R}$ and its inverse are seldomly used to solve the Wiener-Hopf equation [3] because calculating $\tilde{R}$ and its inverse causes numerical instability. Instead, LQ-decomposition with forward- and back-substitution is used. First, the more general solution of the STAP Wiener-Hopf equation is discussed, and then the adaptive beamforming solution (in which a diagonal loading factor is added) is shown.

Compute the LQ decomposition $x_T = LQ$, where L becomes a lower triangular matrix and Q becomes a unitary matrix. The LQ decomposition is related to the QR decomposition described in [2], by inputting the Hermitian of x_T, and receiving the output as the Hermitian of L. Matlab code for the LQ decomposition follows:

```
%lqHouse   Perform LQ decomposition of input matrix
function L = lqHouse(A)
% Find the size of the input matrix A.
[m, n] = size (A);

% Loop over each row of A.
for row = 1:m

% Compute the Householder vector v.
   x = 0;
   x(row:n, 1) = (A(row, row:n))';
   v = x;
   v(row) = v(row) + (x(row) / abs(x(row))) * norm(x);
   beta = - 2 / (v' * v);

% Apply the Householder reflection to the remainder of the
matrix.
   w(row:m, 1) = beta * A(row:m, row:n) * v(row:n);
   A(row:m, row:n) = A(row:m, row:n) + (w(row:m) * v(row:n)');
```

```
end  % for row

L = A;
```

Now the estimated noise covariance matrix can be expressed as:

$$\tilde{R} = (x_T)(x_T)^H = (LQ)(LQ)^H = LQ(Q^H L^H) = LL^H \quad , \tag{17}$$

and the Wiener-Hopf equation can be rewritten as:

$$w = \tilde{R}^{-1} v = (LL^H)^{-1} v \tag{18}$$

$$(LL^H)w = (LL^H)(LL^H)^{-1} v = v \tag{19}$$

Letting $t = L^H w$, $Lt = v$ can be solved for t by using forward substitution since L is lower triangular. Then using $t = L^H w$, a solution for the adaptive filter vector, w, can be found by using back-substitution since L^H is upper triangular.

Now the added complication of the added diagonal loading factor in the adaptive beamforming stage is discussed. To solve this problem, define: $x_T' = \left[x_T \; I\sqrt{\alpha}\right]$. Now $x_T' = L'Q'$, and the Wiener-Hopf equation, $w = ((L')(L')^H)^{-1} v$, can be solved as described above.

Finally, the denominator of equation (4) is a scalar which can be solved in the following manner. Using $\tilde{R} = (L')(L')^H$,

$$v^H \tilde{R}^{-1} v = v^H ((L')(L')^H)^{-1} v = v^H ((L')^H)^{-1} (L')^{-1} v \quad . \tag{20}$$

Letting $u = (L')^{-1} v$, $L'u = v$, which can be solved for u using forward substitution, and $v^H \tilde{R}^{-1} v = u^H u$, which is a scalar.

ACRONYMS

CFAR – Constant False Alarm Rate

CPI – Coherent Processing Interval

FFT – Fast Fourier Transform

FIR – Finite Impulse Response

GMTI – Ground Moving Target Indicator

PCA – Polymorphous Computer Architecture

PRI – Pulse Repetition Interval

SNR – Signal-to-Noise Ratio

STAP – Space-Time Adaptive Processing

REFERENCES

[1] William G. Coate, "Preliminary Design Review: Kinematic Tracking for the PCA Integrated Radar-Tracking Application," MIT Lincoln Laboratory Project Report PCA-IRT-4, 25 February 2003, issued 6 February 2004.

[2] Gene H. Golub and Charles F. Van Loan, *Matrix computations*, 3rd Edition, Johns Hopkins University Press, 1996.

[3] Simon Haykin, *Adaptive Filter Theory*, 4th Edition, Prentice-Hall, Inc., 2002.

[4] Alan V. Oppenheim and Ronald Schafer, *Discrete-time signal processing*, Prentice-Hall, Inc., 1989.

[5] John G. Proakis, Charles M. Rader, Fuyun Ling, and Chrysostomos L. Nikias, *Advanced digital signal processing*, Macmillan Publishing Company, 1992.

[6] Daniel V. Rabinkin and Nicholas B. Pulsone, "Subband-domain signal processing for radar array systems," in *Proceedings of the SPIE*, Vol. 3807, Denver, July 1999.

[7] Daniel V. Rabinkin and Huy T. Nguyen, "Optimum subband filterbank design for radar array signal processing with pulse compression," *Proceedings SAM2000*, Cambridge, March 2000.

[8] Daniel V. Rabinkin and Huy T. Nguyen, "An efficient architecture for a multichannel array subbanding system with adaptive processing," in *Proceedings of the SPIE*, Vol 4116, San Diego, August 2000.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	8 April 2002	Project Report

4. TITLE AND SUBTITLE

Preliminary Design Review: GMTI Processing for the PCA Integrated Radar-Tracker Application

5. FUNDING NUMBERS

C — F19628-00-C-0002

6. AUTHOR(S)

A.I. Reuther

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lincoln Laboratory, MIT
244 Wood Street
Lexington, MA 02420-9108

8. PERFORMING ORGANIZATION REPORT NUMBER

PR-PCA-IRT-2

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DARPA/ITO
3701 Fairfax Drive
Arlington, VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ESC-TR-2003-070

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This report describes ground moving target indicator (GMTI) processing to be done for the PCA (polymorphous computing architecture) integrated radar-tracker application. GMTI processing of the raw radar data is done to extract targets; target reports are passed on to the tracker part of the integrated radar-tracker application.

The GMTI processing described in this report is used to minimize the effects of interference patterns and ground clutter and to recognize and extract targets on the ground. The radar is presumed to be using a wideband signal for improved resolution. This signal is decomposed via subband filtering into many narrowband signals for easier processing. Each of these subbanded data sets must then be processed as a distinct radar data cube. For each data set, adaptive processing is used to null interference patterns and ground clutter noise. These operations incur a high computational cost. Subband filtering requires many parallel modulators and filters, while adaptive processing requires the calculation of an adaptive filter (involving a QR decomposition and forward/backsolving) at run time for each subband data cube.

14. SUBJECT TERMS			15. NUMBER OF PAGES 36
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Same as Report	Same as Report	Same as Report